**学习进度**

本周学习任务：

补充学习注意力机制原理；运行并理解引入自注意力机制的图像识别程序。

1. 什么是注意力Attention?

通俗的说，Attention机制希望网络能够自动地学习：图片或文字序列中需要注意的地方。比如，人眼在看一幅画的时候，不会将注意力平等地分配给画中的所有像素，而是将更多注意力分配给人们关注的地方。

正式的说，Attention机制在给定的位置范围（默认是全局上下文）内，通过加权求和的方式，捕获不同位置之间的关联信息，从而并行重构了不同位置上对应的分量。Attention机制处理的输入序列，可以是由文本中若干个词构成的词向量矩阵，可以是图像处理领域的原始图像，也可以是不同网络层中传递的特征图。

尽管，一篇来自Google的经典论文，告诉我们“Attention is All You Need！”。然而，独立的Attention模型并没有彻底解决忽视位置信息的问题，即使引入了位置编码。此外，并非所有问题都需要全局依赖，一个来源于卷积窗口思想的改进是Truncated Self-attention，即：每次只考虑局部上下文信息进行加权求和。显而易见的是，把Attention作为一个自由模块/网络层，插入到CNN、RNN等模型网络架构的不同位置，是一个扬长补短的明智做法。

1. 自然语言处理领域的Attention

（1）传统方法

在自然语言处理领域，常规的做法是基于递归框架来处理词向量矩阵，比如RNN、LSTM、GRU及SRU等循环序列模型。此类方法本质上是马尔科夫决策过程，都无法很好地学习到全局上下文信息。此外，更重要的问题是：循环序列模型需要获取浅层输出的激活值，才能依序进行更深层的处理，显著缺乏并行能力。

另一种有趣的做法是，利用CNN的窗口式遍历思想来处理词向量矩阵，从而通过层叠的方式来逐步扩大感知域，捕获不同层次的局部信息。此外，在每个卷积层中使用多个卷积核进行卷积操作，也一定程度上彰显了并行能力。值得注意的是，使用TextCNN等卷积神经网络变体，处理文本数据时所采用的窗口式遍历思想，恰好同注意力机制原理不谋而合。

（2）基于注意力机制的方法

如果需要一步到位地获取全局信息，Attention模型则呼之欲出：

若A=B=X，企图寻找序列内部的联系，则为Self-Attention。自注意力作为常见的基础实例，公式为：。其中，用于计算不同分量之间的相似程度（计算方法有点积、拼接及感知机等，此处为点积）；因子用于控制内积值不至于太大，防止softmax归一化后的结果非0即1。自注意力通过不同分量的Q与所有分量的K进行匹配，以获取所有分量对于当前位置的重要程度（注意力权重），从而将给定范围内的分量V作为特征输入，通过加权求和的方式重构当前位置的信息。

若需要获取不同类型的语义关联性，以提高模型的表达能力，则Multi-Head Attention应运而生。多头注意力机制，类似于CNN中一个卷积层使用多个卷积核的思想，可以抽象地理解为：“多做几次同样的事，然后将结果拼接”。具体而言，每个位置上的分量都对应有多个不同的Q、K、V，每一组Q、K、V都独立进行自注意力权重计算，即：。对于每个位置而言，还需要将当前位置对应的注意力权重拼接起来，即：。

自注意力只考虑到了语义关联性，忽略了词序信息，即：变换序列中不同分量的位置，获取的注意力权重仍然一致。为此，可以将序列中每个位置上的分量与对应的位置向量相加/拼接，从而通过引入位置编码的方式，考虑到词序信息。问题是：如何获取对应位置的位置编码？常见的做法是基于公式或通过训练来获取位置编码，有研究表明：两种方法获取的位置编码效果接近。由于位置编码本身表达了绝对位置的信息，而基于公式的方法额外提供了表达相对位置信息的可能性，因此基于公式来映射获取位置编码的方式更为简便可取。

**小结：**

同其它传统模型相比，可知：RNN要依序递推才能捕捉上下文联系，且存在长期记忆遗失问题；CNN通过层叠来逐步扩大局部感知能力，存在并行能力；而Attention能够一步到位地捕捉给定范围内的全局联系，善于进行长距离依赖学习且更具并行性。

值得注意的是，“Attention is All You Need”是一种较为理想化的想法。比如在翻译任务中，由于词序信息不太重要，自注意力模型某种程度上可以忽略本身缺乏考虑位置信息的缺点，但是词性标定这一类更需要考虑到词序信息的任务呢？引入位置编码确实是一种可行的办法，然而通过在序列上相加或拼接位置向量的方式，本身就存在信息损失及位置编码可靠性的问题。如前所述，根据任务情景及实际效果，把Attention作为一个自由模块/网络层，插入到CNN、RNN等模型网络架构的不同位置，是一个扬长补短的明智做法。

1. 计算机视觉领域的Attention

在计算机视觉领域，注意力机制通过学习权重分布的方式，表象上控制了图像中不同位置所对应的专注度。如前所述，Attention处理的输入序列可以是原始图像，也可以是网络中传递的特征图。同样，重构当前位置信息的依据可以是全局分量，也可以是以某种采样策略选取的局部分量。此外，你既可以在空间尺度上引入注意力（SAM），也可以在通道尺度上引入注意力（SENet），甚至可以在两个尺度上同时引入注意力（CBAM）。显然，在计算机视觉领域的各类任务情景中，注意力机制还能作为一个自由的组件，嵌入到CNN等模型网络架构的不同层之间。具体该怎么做呢？实验如下：

**（1）实验说明**

① 实验环境：基于Python3.9内核的Miniconda Jupyter Notebook

② 开发框架：torch2.1.0+cu121

③ 实验目的：识别模糊图片中物体的类别（多分类任务）

④ 模型架构：引入空间注意力模块的卷积神经网络CNN

⑤ 数据集：CIFAR-10

✸ 图片标签：10个类别（飞机、汽车、鸟、猫、狗、鹿、青蛙、马、船、卡车）

✸ 图片规格：32\*32\*3维度的模糊图片，训练集60000张，预测集10000张

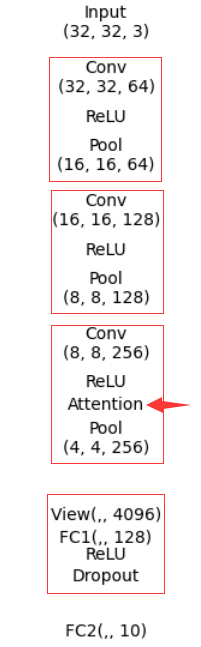
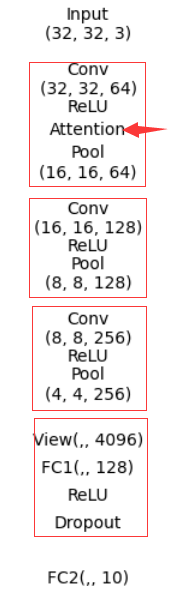
1. **模型架构**

如下图所示，本次实验采用的CNN模型网络架构，由三组Conv+ReLU+Pool、两层全连接层FC、一层Dropout及一层自由插入的Attention层构成。

其中，自定义Self-Attention模块（class SelfAttention(nn.Module)）的实现逻辑如下：

在初始化函数\_\_init\_\_中：首先，创建三个卷积层(query\_conv, key\_conv, value\_conv)用于将X映射为查询Q、键K和值V；然后，定义一个可学习参数gamma，用于调整注意力输出和输入的比例。

在前向传播函数forward中：首先，将输入数据X映射为Q、K、V；然后，依赖Q与K计算注意力能量energy，并归一化以获取注意力权重；最后，依赖注意力权重对V进行加权求和，并通过残差连接将结果与原始输入X相加，从而得到最终的输出。

1. **模型训练与测试**

在进行模型训练及测试前，考虑到实验设备的硬件配置，事先指定GPU作为运算负载。此外，还需要初始化网络模型实例、损失函数（CrossEntropyLoss）及优化器（随机梯度下降SGD）。

在基于mini-batch的模型训练迭代过程中，每次迭代都需：重置损失记忆器及梯度信息、投喂批次数据并计算损失值、通过反向传播计算损失相对于参数的梯度，并更新可学习参数、计算当前批次数据的平均损失。

在模型测试的过程中，还需要额外计算平均准确率，实现方式如下：

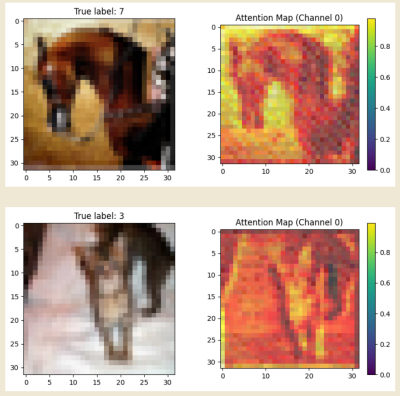
total += labels.size(0)

correct += (predicted == labels).sum().item()

print('Test Accuracy: {:.2f}%'.format(100 \* correct / total))

1. **可视化注意力权重**

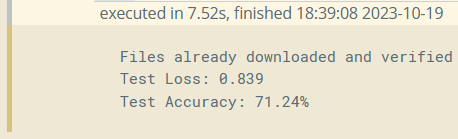
本环节选取2个样本的原图及标签，并通过训练好的模型获得对应的注意力权重矩阵。针对每个样本，通过matplotlib创建一个包含有两个子图的窗口，第一个子图显示原始输入图像及其标签，第二个子图显示注意力权重热力图。可视化注意力权重的对比图，如下所示：



1. **实验结果及说明**

✸ 实验结果

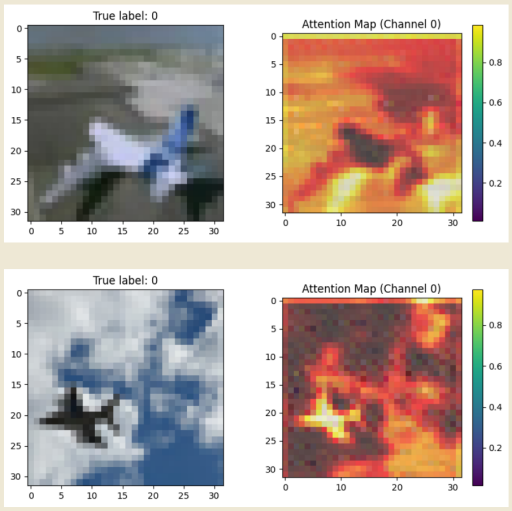
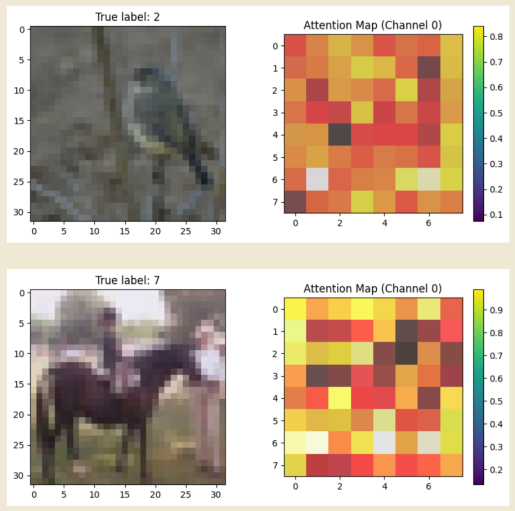
在识别10000张模糊图片中物体类别的测试中，平均损失：0.839，准确率：71.24%。



✸ 情况说明

本次实验中，将空间注意力模块分别放置到第一层卷积层后及第三层卷积层后，在测试集上的准确率分别是：71.24%及71.57%，测试结果差别不大。

然而，将注意力模块放置到第一层卷积层后的，注意力权重可视化图，如下左图所示。将注意力模块放置到第三层卷积层后的，注意力权重可视化图，如下右图所示。

显而易见的是，由于浅层网络中的特征图仍保有原始图像中大量的初始特征，使得处理浅层特征图的注意力模型，所生成的注意力权重热力图，仍能模糊体现物体轮廓等低级特征信息。随着网络层次的加深，CNN逐步扩大的感受域，将从大量原始特征中降维提取更为抽象复杂的高级特征。这使得处理深层特征图的注意力模型，往往更强调高级特征中的突出部分，自然不会在注意力权重热力图中显现物体轮廓等较为低级的特征信息。值得注意的是，存在网络资料说明，此类情况也并不绝对。在一些特定任务中，取决于任务需求和设计者的选择，自定义的注意力模块也可以用来突出显示特定的低级特征。